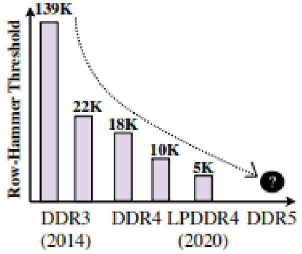
Aggressor-focused Mitigation: exploration of row-migration (RRS/SRS) and enhancements

Hena Naaz

CS7292 Project Presentation

Introduction

What problem are we solving?



Row-Hammer Threshold over time

Row Hammer: security issue that can exploit DRAM memory modules causing bit flips. The decreasing threshold over time has made the systems more vulnerable to attacks. There are various victim-focused mitigative approaches however they fail as they preserve spatial connection between victim and aggressor.

Aggressor-focused approaches: RRS/SRS performs aggressor **row migration** to random address, when a specific activation threshold is attained. However, due to random nature of address, it is still susceptible to attacks.

Prior Work

RRS/SRS:

Aggressor-focused memory-level defense mechanism against RHA that randomly shuffles the location of rows within a memory chip.

SRS divides memory rows into groups and shuffles only within the group, ensuring that sensitive data remains within its designated group.

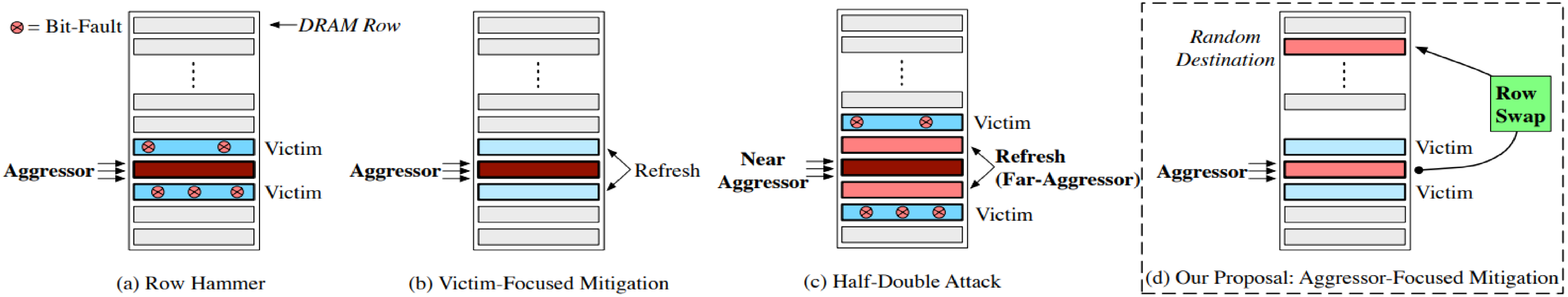


Figure 1: (a) Classical Row Hammer attack (b) Victim-focused mitigation refreshes immediate neighbors (c) Half-Double attack breaks victim-focused mitigation (d) Randomized Row-Swap breaks spatial connection between aggressor and victim row.

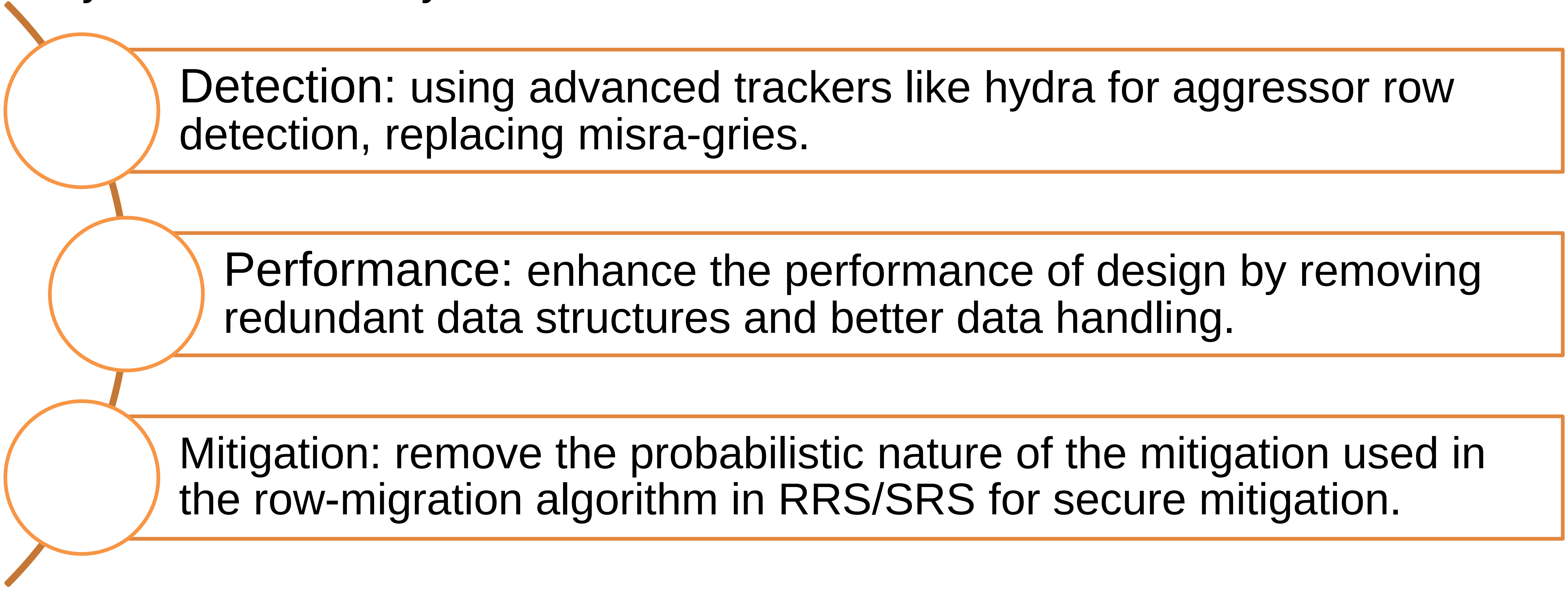
Shortcoming of Prior Work

- Persisting possibility of side-channel attacks in probabilistic model, which can exploit the memory access patterns of a system to infer sensitive information.
- With decreasing threshold, the overall slowdown and SRAM overhead increases significantly.
- Large number of migrations (2 reads, 2 writes)

Not performance efficient/foolproof!
(simulated for observations)

Your Insight

Objective: The goal is to improve the architecture by exploring the optimization on detection, mitigative action and overall performance in terms of cycles and memory overheads.



Detection: using advanced trackers like hydra for aggressor row detection, replacing misra-gries.

Performance: enhance the performance of design by removing redundant data structures and better data handling.

Mitigation: remove the probabilistic nature of the mitigation used in the row-migration algorithm in RRS/SRS for secure mitigation.

Proposal (A & B)- RRS/SRS Enhancement

A: RRS/SRS Design Enhancement using Hydra Tracker

- The design was extended for the row indirection table and swap counters using hydra tracker for storing rows status using a swap group manager to reduce overhead of swap process during row migration.

B: Design enhancement with integration of row tracker and row indirection table

- The goal was an improved performance and scalability of the RRS/SRS technique by combining HRT and RIT to reduce accesses and lookup.

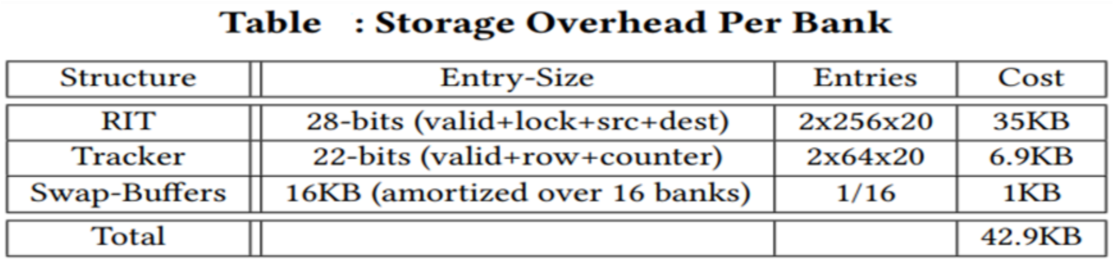


Table : Storage Overhead Per Bank

Structure	Entry-Size	Entries	Cost
RIT	28-bits (valid+lock+src+dest)	2x256x20	35KB
Tracker	22-bits (valid+row+counter)	2x64x20	6.9KB
Swap-Buffers	16KB (amortized over 16 banks)	1/16	1KB
Total			42.9KB

Proposal - (C) Non-random row-migration

Based on AQUA that uses a quarantine region with forward and reverse pointers for the aggressor rows being pushed into the quarantine region.

The enhancement used the Hydra tracker for ART in this design and leveraged victim refresh and row-migration hybrid for mitigation.

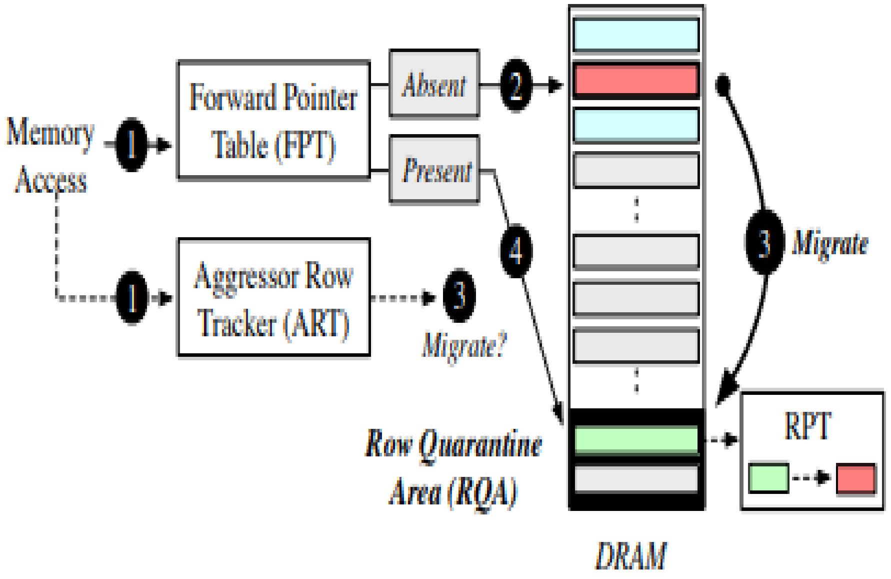


Figure Overview of AQUA. The Forward-Pointer Table (FPT) determines if the access should go to the original or the quarantined location. The Aggressor-Row Tracker (ART) identifies rows that must get quarantined.

Methodology

Criterion	Used methodology
Simulator	Simple Sim (CS7292 Lab1 simulator)
Benchmarks	(SPEC) bzip, bwaves, Gems, cactus, mcf, zeusmp
Configuration	4 CPU OoO cores and multi-level set-associative caches
Memory config	1 Channel * 1 rank * 16 banks Memory Size: 16 GB, Row Size: 8 KB
Metrics	Number of migrations, total sys cycles, mitigations, total activations, number of entries in tables, memory overhead.

Key Results (A & B)

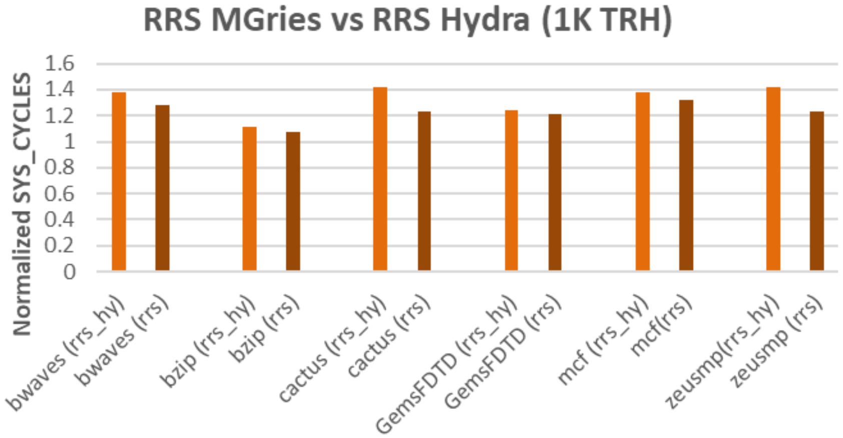
Proposal A: SRAM based tracker with RRS/SRS

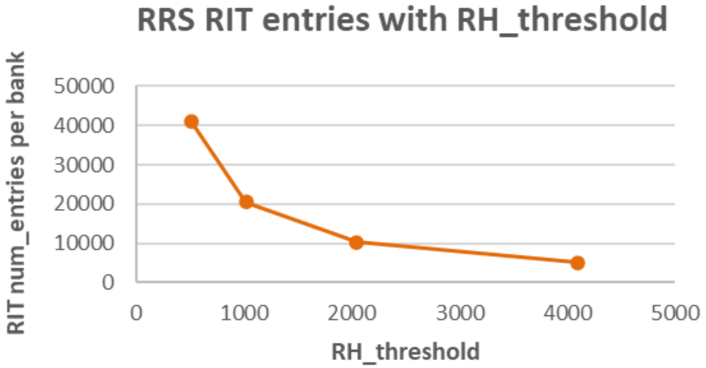
Security: Not secure since random migration is probabilistic!

Performance: Minor cycle improvement but no significant other metrics with Hydra!

Motivation:

Structure	RRS-MG	RRS-Hydra
Tracker	396 KB	28.3 KB
Mapping Tables-RIT	2.4 MB	2.4 MB
Swap Buffers	16 KB	16 KB
Total	2,870 KB	2,502 KB





Proposal B: RRS/SRS RIT design enhancement

Security: No security improvement! As it is still probabilistic.

Performance: increased storage overhead.

HRT becomes very bulky with swap pairs, and the amount of lookups doubles.

Experiment:

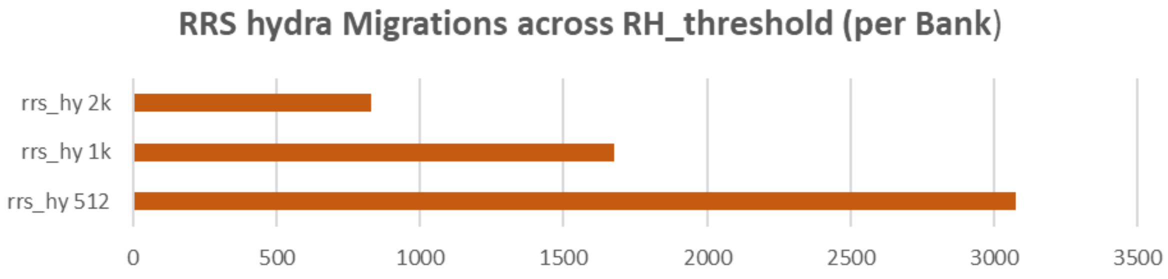
Design Enhancement	Entry per Bank
Independent HRT RIT	42.9KB
Combined HRT RIT	45.8KB

Key Results: (C) AQUA hydra Gains

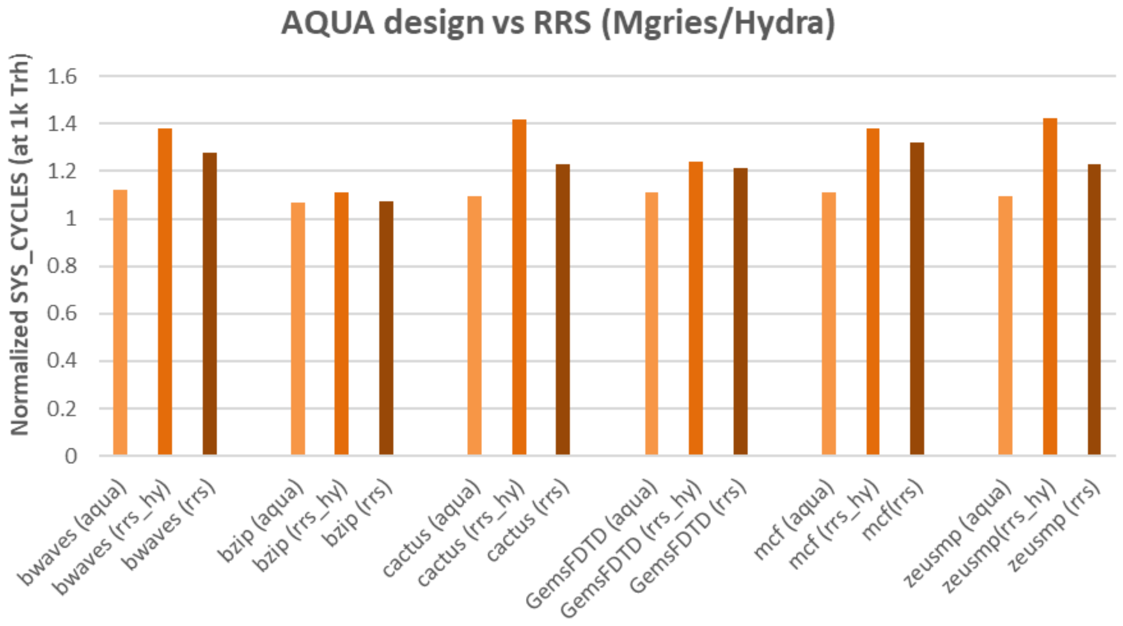
Major gains with Aqua hydra vs RRS hydra:

Motivation:

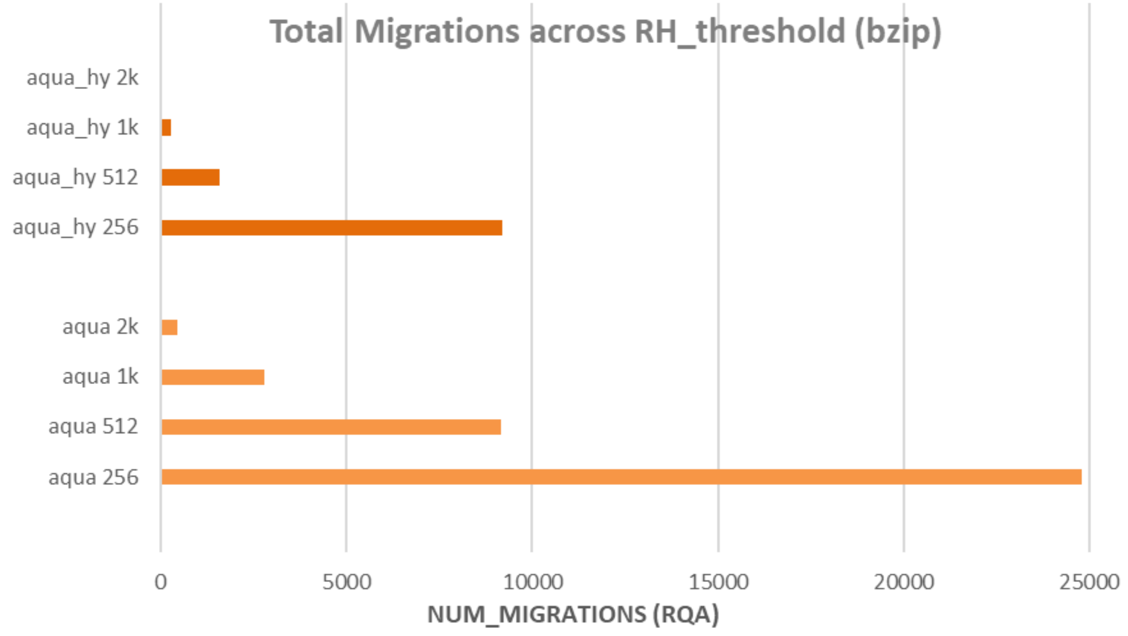
Structure	**AQUA-MG**	**AQUA-Hydra**
Tracker	396 KB	30.3 KB
Mapping Tables	32.6 KB	32.6 KB
Buffer	8 KB	8 KB
Total	437 KB	71 KB



RRS hydra Migrations across RH_threshold (per Bank)

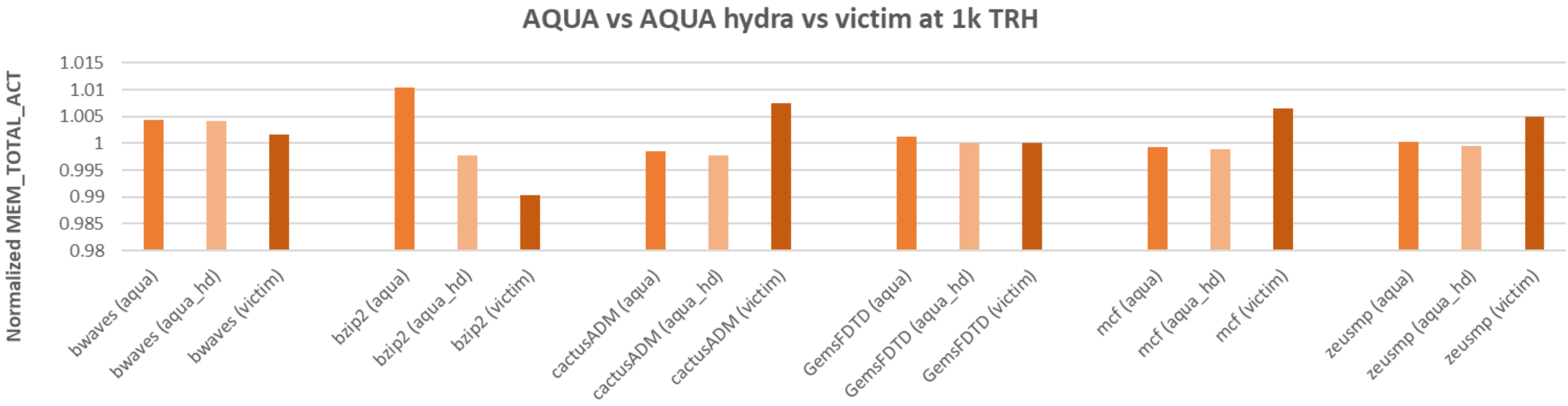


AQUA design vs RRS (Mgries/Hydra)

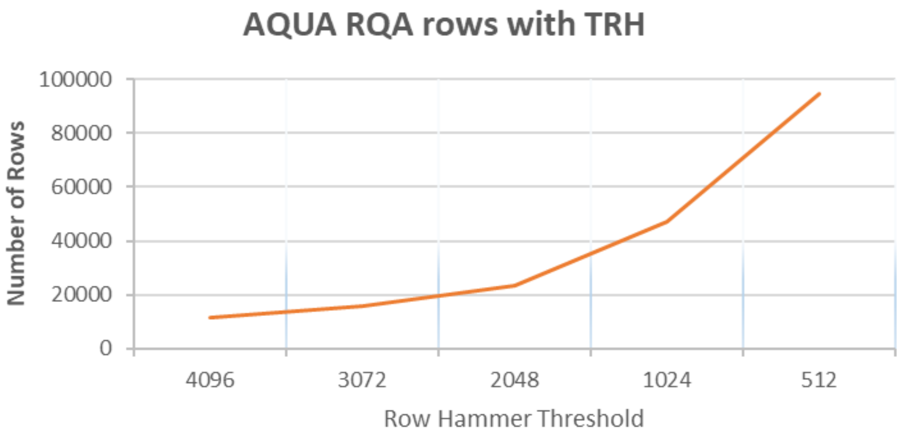


Total Migrations across RH_threshold (bzip)

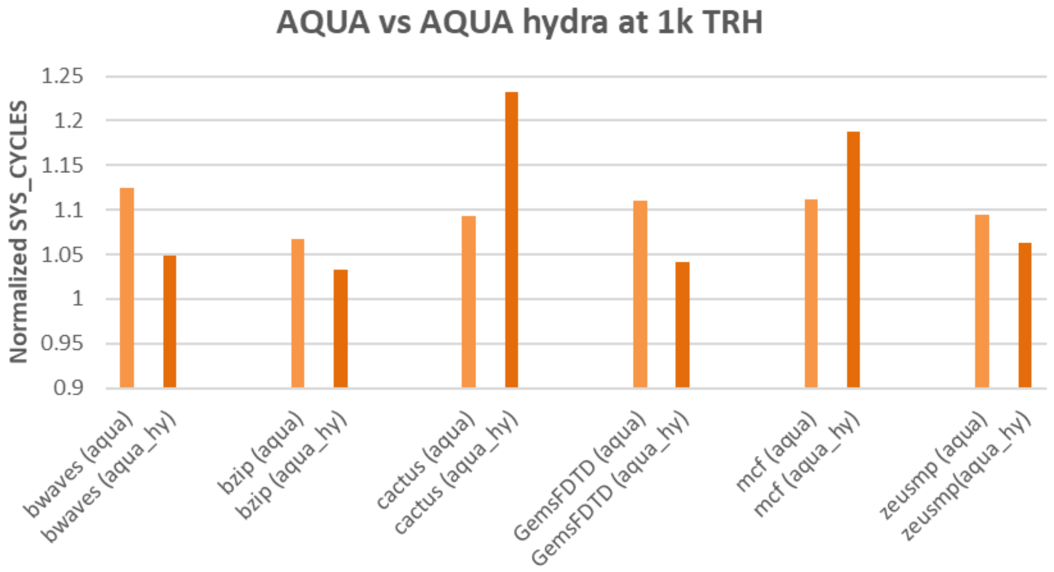
Key Results: (C) AQUA hydra Overheads



AQUA vs AQUA hydra vs victim at 1k TRH



AQUA RQA rows with TRH



AQUA vs AQUA hydra at 1k TRH

Conclusion

- Unlike RRS and SRS, AQUA does not rely on randomization (potential security threat), rather isolation(quarantine).
- By integrating the Hydra tracker and modifying the aqua design for aqua, we improved the efficiency of design in terms of storage overhead.
- Experiments showed that AQUA hybrid with non-random row-migration and enhancements also outperformed RRS and SRS in terms of the number of row migrations and performance as the total cycles reduced considerable compared to RRS with hydra.

Source code: https://github.com/henanaaz/RSCA_Project.git

References:

- Gururaj, Saileshwar et al. “Scalable and Secure Row-Swap: Efficient and Safe Row Hammer Mitigation in Memory Systems.” (HPCA), 2023.

- Saxena, Anish et al. "AQUA: Scalable Rowhammer Mitigation by Quarantining Aggressor Rows at Runtime." (MICRO), 2022.

- Qureshi, Moin et al. "Hydra: Enabling Low-Overhead Mitigation of Row-Hammer at Ultra-Low Thresholds via Hybrid Tracking." (ISCA), 2022.

- Gururaj, Saileshwar et al. “Randomized Row-Swap: Mitigating Row Hammer by Breaking Spatial Correlation between Aggressor and Victim Rows.” (ASPLOS), 2022.